AMENDMENTS TO THE CLAIMS

This listing of the claims will replace all prior versions and listings of claims in the application. Claims 40, 45, and 50 are amended herein. Claims 1-39, 42-43, 47-48 and 52-53 remain canceled herein without prejudice. No new claims are added.

Listing of Claims:

1-39. (Cancelled).

40. (Currently amended) A method for modifying an application to provide functionality for tracing a program flow of the application at a user-configurable level of granularity specified via a Graphical User Interface presented at an end-user device, the method comprising:

reading program code from memory and processing said program code with one or more processors to perform the following:

presenting to the end-user device via the Graphical User Interface, options for modifying the application's bytecode by injecting tracing and debugging operations into the application's bytecode at the user-configurable level of granularity specified via the Graphical User Interface, wherein the application is composed of a plurality of archive files associated with two or more distinct tiers in a multi-tiered architecture, the archive files having respective class files, and the respective class files having respective methods, and wherein the options for modifying the application's bytecode includes:

modifying bytecode of only a subset of a plurality of methods from which the application is composed, the subset of the plurality of methods selected from two or more class files in two or more archive files composing the application's bytecode as specified via the Graphical User Interface presented to the end-user device, wherein the two or more

archive files are associated with two or more distinct tiers of the multi-tiered architecture, and wherein the modified subset of the plurality of methods specified provides the user-configurable level of granularity by providing the functionality for tracing the program flow of the application through only the subset of the plurality of methods specified via the tracing and debugging operations injected into the subset of the plurality of methods specified;

executing the application in an object oriented runtime frame work, wherein executing the application includes processing a portion of the application's bytecode that was modified in accordance with the user-configurable level of granularity specified via the Graphical User Interface; [[and]]

presenting output generated from execution of the portion of the application's bytecode that was modified to the end-user device via the Graphical User Interface;

wherein the portion of the application's bytecode that was modified includes a method entry or method exit;

wherein said dispatch unit is invoked by the modified bytecode and recognizes the invoking method associated with the modified bytecode by way of receiving the invoking method's class identifier, the invoking method's identifier, and the invoking method's parameters;

wherein the dispatch unit redirects program flow to one or more plug-in modules comprising plug-in handlers; and

wherein said output is provided by said plug-in handlers.

41. (Previously Presented) The method of claim 40 wherein said object oriented runtime framework is a Java compliant object oriented runtime framework.

42-43. (Canceled)

44. (Previously Presented) The method of claim 40 wherein said output includes one of:

a time at which said method entry or method exit was entered; and,

a parameter that is passed at said method entry or method exit.

45. (Currently amended) A machine readable storage medium containing program code that when processed by one or more processors of a computer causes a method to be performed:

presenting to an end-user device via a Graphical User Interface, options for modifying the application's bytecode by injecting tracing and debugging operations into the application's bytecode at a user-configurable level of granularity specified via the Graphical User Interface, wherein the application is composed of a plurality of archive files associated with two or more distinct tiers in a multi-tiered architecture, the archive files having respective methods therein, and wherein the options for modifying the application's bytecode includes:

modifying bytecode of only a subset of a plurality methods from which the application is composed, the subset of the plurality of methods selected from two or more archive files composing the application's bytecode as specified via the Graphical User Interface presented to the end-user device, wherein the two or more archive files are associated with two or more distinct tiers of the multi-tiered architecture, and wherein the modified subset of the plurality of methods specified provides the user-configurable level of granularity by providing the functionality for tracing the program flow of the application through only the subset of the plurality of methods;

executing the application in an object oriented runtime frame work; [[and]]

presenting output generated from execution of the portion of the application's bytecode that was modified to the end-user device via the Graphical User Interface;

wherein the portion of the application's bytecode that was modified includes a method entry or

method exit;

wherein said dispatch unit is invoked by the modified bytecode and recognizes the invoking method associated with the modified bytecode by way of receiving the invoking method's class identifier, the invoking method's identifier, and the invoking method's parameters;

wherein the dispatch unit redirects program flow to one or more plug-in modules comprising plug-in handlers; and

wherein said output is provided by said plug-in handlers.

46. (Previously Presented) The machine readable storage medium of claim 45 wherein said object oriented runtime framework is a Java compliant object oriented runtime framework.

47-48. (Canceled)

49. (Previously Presented) The machine readable storage medium of claim 45 wherein said output includes one of:

a time at which said method entry or method exit was entered; and,

a parameter that is passed at said method entry or method exit.

50. (Currently amended) A computer comprising a Graphical User Interface to present options for modifying an application's bytecode by injecting tracing and debugging operations into the application's bytecode at a user-configurable level of granularity specified via the Graphical User Interface, wherein the application is composed of a plurality of archive files associated with two or more distinct tiers in a multi-tiered architecture, the archive files having respective methods therein, and wherein the options for modifying the application's bytecode includes:

modifying bytecode of only a subset of a plurality of methods from which the application is

composed, the subset of the plurality of methods selected from two or more class files in two or more archive files composing the application's bytecode as specified via the Graphical User Interface, wherein the two or more archive files are associated with two or more distinct tiers of the multi-tiered architecture, and wherein the modified subset of the plurality of methods specified provides the user-configurable level of granularity by providing the functionality for tracing the program flow of the application through only the subset of the plurality of methods specified; [[and]]

the Graphical User Interface to further present output generated from execution of the portion of the application's bytecode that was modified;

wherein the portion of the application's bytecode that was modified includes a method entry or method exit;

wherein said dispatch unit is invoked by the modified bytecode and recognizes the invoking method associated with the modified bytecode by way of receiving the invoking method's class identifier, the invoking method's identifier, and the invoking method's parameters;

wherein the dispatch unit redirects program flow to one or more plug-in modules comprising plug-in handlers; and

wherein said output is provided by said plug-in handlers.

51. (Previously Presented) The computer of claim 50 wherein said object oriented runtime framework is a Java compliant object oriented runtime framework.

52-53. (Canceled)

54. (Previously Presented) The computer of claim 50 wherein said output includes one of:

a time at which said method entry or method exit was entered; and,

a parameter that is passed at said method entry or method exit.